Physica 12D (1984) 396–407
North-Holland, Amsterdam

3.2. COMPUTATIONAL METHODS

NUMERICAL METHODS FOR TRACKING INTERFACES*

James M. HYMAN
Center for Nonlinear Studies, Theoretical Division, MS B284, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

An overview is given of special numerical methods for tracking discontinuous fronts and interfaces. These methods include surface tracking methods based on connected marker points along the interface, volume tracking methods that track the volume occupied by the solution regions bounded by the interfaces, and moving-mesh methods where the underlying mesh is aligned and moved with the interface. The pros and cons of the current methods are discussed and a new method is proposed that overcomes some of the difficulties encountered in approximating equations with multiple interacting interfaces.

1. Introduction

Interface tracking methods are often necessary to efficiently compute accurate numerical approximations to partial differential equations with moving discontinuous interfaces in the solution. There are a few well-established algorithms that account for these discontinuities, but most are numerical schemes† still in their early developmental stages. No simple rules exist for choosing the best method for the more difficult problems. In this paper I will give an overview of the current methods in order of their ability to handle problems of increasing difficulty. I will then introduce a new adaptive moving-mesh scheme and speculate on what will be the more significant future developments.

Much of our understanding of the laws of nature is based on integral equations and constitutive relationships that hold across discontinuous interfaces. Away from these discontinuities, where the solution is smooth, these equations can be well approximated by partial differential equations (PDEs). Most numerical predictions of the laws of nature are based on discrete approximations to these PDEs. For these numerical methods to be accurate near discontinuities, where the PDEs fail to approximate the integral equations, they must treat the discontinuity as a special case. Otherwise, the method may not accurately approximate the physically relevant solution.

Two commonly used approaches are to smear the interface by adding artificial dissipation or viscosity to the PDEs and solving this nearby problem, or to treat the discontinuity as an internal boundary, solving the PDEs away from the discontinuity and imposing the appropriate jump conditions across this boundary.

The better artificial dissipation methods are extremely easy to implement, perform excellently for a restricted class of problems, but are not well understood theoretically. One of the more important problems they cannot treat adequately is tracking a moving internal material interface or slip line when the equation-of-state for the two materials is radically different. For example, the water/air interface of a bubble should not be artificially smeared, or the method will not accurately account for the effect of surface tension on its motion.

Another difficulty with the artificial dissipation method occurs when it is used in conjunction with an adaptive static rezone method [2, 18]. These

*This work was supported by the U.S. Department of Energy under contract W-7405-ENG-36 and the Office of Basic Energy Sciences, Department of Applied Mathematics under contract number KC-04-02-01-0.

†Scheme – (def): an especially sly or devious plan of action. The Random House Dictionary of the English Language, (Random House, New York, 1967).

adaptive methods automatically adjust the mesh so that it is dense in regions with sharp transitions and sparse where the solution is smooth. The artificially smeared discontinuities are likely to have the large gradients causing the adaptive mesh algorithm to introduce mesh points that resolve the structure of the transition layer. This structure is often an artifact of the artificial dissipation and does not significantly affect the behavior of the solution. (If it did, then the artificial dissipation method would not have been appropriate.) These extra mesh points can greatly increase the computational expense without significantly enhancing the accuracy of the calculation.

The interface tracking methods described in this paper were developed to overcome the deficiencies of the artificial dissipation approach. The tracking methods have little or no artificial dissipation near the interface since the singularity is directly computed and treated explicitly as a discontinuity. These methods are more difficult to implement, perform excellently for a large class of problems, but, like artificial dissipation methods, are not well understood theoretically for the more difficult problems.

Interface tracking methods can be divided into three catagories. In order of increasing flexibility and computational complexity, these are surface tracking methods, volume tracking methods, and moving mesh methods.

Surface tracking methods track the location of the interface by interpolating between marker particles along the interface. Because this is a lower dimensional problem, the additional effort to accurately resolve small-subgrid-scale structure in the interface is usually small compared with the overall solution time. The surface tracking methods are the simplest to implement, until interactions occur that change the topology of the interface during the computation.

Volume tracking methods overcome the changing topology problems by dividing the domain into a union of disjoint solution regions. The boundary between these regions is the interface location. The regions are identified by marker points, or alternatively, the fractional volume of each solution region located in each computational cell is calculated and advanced during the computation by solving an auxiliary evolutionary PDE. These fractional volumes can be used to reconstruct an approximate interface location at any time. Unlike the surface tracking methods, very little subgrid scale structure is retained during the calculation.

Moving-mesh methods can be used to track the location, account for changes in the interface topology, and resolve small-scale structures in the interfaces. Here a multivalued solution is defined at mesh points located on the interface. The interface mesh points move with the interface in a Lagrangian manner. To prevent the mesh from tangling, a dynamic data structure is used so the moving mesh points can change their nearest neighbors during the calculation. Also, new points are added when the mesh becomes sparse or a new interface appears, and mesh points are removed when they are more dense than necessary for an accurate calculation.

In all the above methods, the location of the interface is advanced by solving a lower dimensional PDE derived from an appropriate constitutive jump condition. The effect of the interface movement is transferred to the solution to the original PDEs on either side where the interface is treated as a moving boundary.

2. Surface tracking methods

In surface tracking methods the interface is specified by an ordered set of marker points located on the interface [11, 12, 22, 25]. Between these points its position is approximated by an interpolant, usually a piecewise polynomial. These time-dependent interfaces divide the problem domain into connected regions. The solution defined at the marker points and along the interpolated interface may be multivalued to account for discontinuities.

2.1. *Surface representation*

The marker points may be represented by the distance from some reference surface such as the chain-of-line segments defined by a height function (fig. 1a) or by a parametric interpolant as shown (fig. 1b).

The distance function is simpler to implement, but the interface deformation is severely limited because this representation breaks down if the curve becomes multivalued with respect to the reference surface. The parameter representation does not have these limitations and is only slightly more complicated to implement.

Both of these methods can provide the fine resolution and detail needed to track small-sub-grid-scale structures in the interface in two and three space dimensions. This can be especially important when tracking an unstable interface. The underlying computational grid resolution is usually chosen to resolve the structure of the smooth solution away from the interface and is rarely sufficient to resolve the fine-scale interface detail, such as the onset of a slip-line Helmholtz instability rollup.

The surface tracking methods are hybrid numerical schemes – splitting the solution process into two parts: the interface tracking and the smooth solution algorithm. Both the smooth solution and the interface are treated as separate computational objects. The interface position is stored and dynamically updated along with the smooth solution away from the interface. The numerical method may be implemented to do this simultaneously, or operator splitting may be used, first advancing one and then the other.

2.2. *Surface evolution*

The evolution equations for the interface are lower dimensional differential equations derived from constitutive relationships, usually obtained by applying the divergence theorem to an integral formulation of the physical model. Similar methods are used to define an accurate approximation to the smooth solution near an interface. This approach, sometimes called the finite volume method, results in equations for the interface–not for the marker points. In practice, however, it is the marker points that are evolved. Because of this, extra' care is needed to maintain the appropriate relationships, such as the conservation laws, near the interface.

When the underlying PDE system is hyperbolic, the equations for the interface can be derived by solving one-dimensional Riemann problems nor-

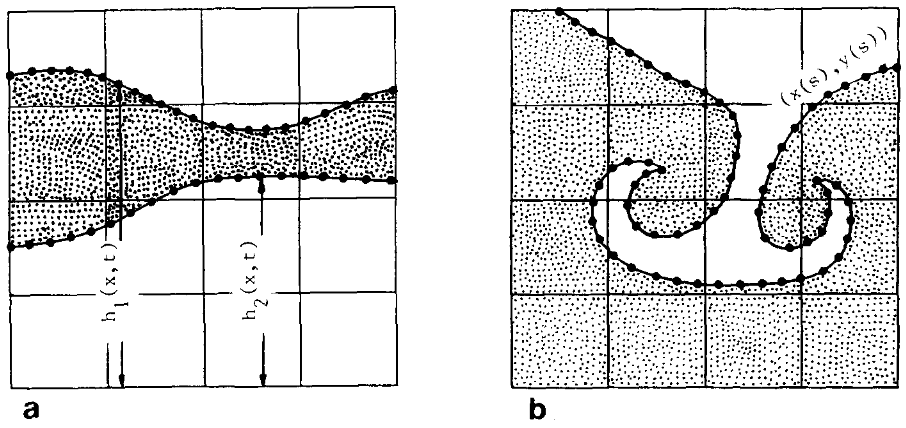

Fig. 1. a) Multiple distance functions designate the locations of the interfaces. b) A parametric interpolant designates the locations of the interfaces.

mal to the interface [11–13]. A Riemann problem is a type of a nonlinear normal mode expansion for a one-dimensional Cauchy problem with initial data that is constant everywhere except for a single jump discontinuity. The solution evolves into nonlinear waves that propagate coherently in time. For example, a material interface discontinuity moves with the underlying fluid velocity, and a shock wave moves with the speed given by the Rankine–Hugoniot jump conditions. When used to advance a 2-D interface in the normal direction, curvature effects can be incorporated as source terms.

When multiple waves are emitted in the Riemann problem, then a single wave, such as the contact discontinuity corresponding to a material interface, must be selected and tracked at all the points along the same interface. Alternatively, additional interfaces can be inserted to track the new discontinuities that arise.

The constant states used for the Riemann problem are the bounding states on either side of the interface. A slightly improved version could be implemented by using a linear or quadratic variation in the states normal to the front, extending the ideas of van Leer [28]. The velocities for the marker points are taken to be the tracked wave velocities normal to the interface. The tangential velocities of the underlying solution also contribute to the interface movement, but are less important, since when the marker points are displaced tangentially, they remain on the interface.

In addition to the Riemann problem equations, the evolution equations may contain additional terms to approximate the effects of surface tension [15, 21], flame propagation [1, 3, 26] or phase changes. These all influence the boundary conditions imposed on the smooth solution at the interface. For example, if the interface were a flame front that converts the fuel ahead of the interface to burned material behind it, the interface boundary conditions would be conservative outflow or inflow conditions, respectively, with the appropriate heat source term to account for the energy released at the interface.

The above procedure is simplified if a local curvilinear coordinate system orthogonal to the interface is employed. The orthogonal coordinate system also simplifies the interface boundary conditions that connect the smooth solution to the interface. Additionally, when approximating the spatial derivatives, we can smoothly map this coordinate system onto a fixed regular computational grid [19, 24].

Sometimes the evolution of the interface is sensitive to small amounts of noise in the computed solution on either side, and it may be necessary to regularize or smooth the solution states in the tangential direction. Glimm, Marchesin, and McBryan [11, 12] average in a circle about each marker point on the interface to filter out short wavelength fluctuations. Another way to reduce the effects of small errors on the motion of the interface is to modify the interface evolution equations slightly by adding artificial surface tension along the interface.

Instead of solving Riemann problems, the vortex methods [3, 26] introduce small lines on surfaces of vorticity along the interface. The vortex motion is defined by a Hamiltonian system of ordinary differential equations with a Coulomb-type interaction term. For viscous PDEs, these equations also contain a diffusive term.

The contour dynamics method of Overman and Zabusky [25] follows the motion of point vortices along a closed contour line bounding a region of constant density or vorticity. The contour velocity is obtained from boundary integral equations derived by applying Green's theorem to the area integral in the Green's function solution of a Poisson equation.

2.3. *Interpolants*

The accuracy of the surface tracking methods depends strongly on stability and accuracy of the interpolation method approximating the interface location between the marker points. The shape-preserving Hermite piecewise-polynomial interpolants [16] remain consistently well behaved and smooth by retaining the convexity and monotonicity properties of the original data. Therefore, extraneous bumps or wiggles are not introduced between the data points. This is particularly important for unstable interfaces such as a Rayleigh–Taylor or Buckely–Leverett interface.

The piecewise-linear interpolant is the simplest shape- and area-preserving interpolant, but introduces fictitious corners in the interface location. In fluid flows, unless the solution is artificially smoothed, these corners can create unrealistic velocities in the smooth solution that eventually destroy the global accuracy of the calculation. The higher order Hermite piecewise-polynomial interpolants are smoother but can still easily accommodate necessary corners in the interface by retaining left and right derivatives at the marker points. Usually, cubic piecewise polynomials are sufficient.

As the interface grows or shrinks, the distribution of the interface marker points must change to continually resolve the interface. The static rezone methods [18] use a mesh function or performance index based on the arc length and curvature of the interface as a guide when removing extraneous mesh points or adding new ones when the existing points are too dense or sparse. Alternatively, a fixed number of marker points can be continuously redistributed so as to best resolve the interface. The first approach is usually more efficient in 2-D calculations where the marker points are easily reordered as points are added or deleted. The data structure is simpler in the second approach and it is more commonly used in 3-D calculations.

2.4. *Interactions*

In many calculations the topology of the interfaces is constantly changing. The interfaces can interact with each other, spontaneously disappear, or new ones can be created when an existing interface bifurcates or is formed by a compression wave, a flame ignition, or a phase change. Multiple interaction points where three or more interfaces meet are common occurrences and also require special data structures.

Fortunately, the interactions are usually a lower dimensional event; in 2-D calculations the 1-D interfaces intersect at points; in 3-D, the 2-D surfaces interact along 1-D lines. Therefore, it takes little computer time to identify and track the interactions. Unfortunately, the data structure and algorithms needed by surface tracking methods to account for interactions greatly increase the program complexity. Also, near the interaction many of the lower dimensional interface evolution equations based on 1-D Riemann problems are no longer valid. That is, near an interaction point, the solution often is intrinsically 2-D or 3-D and the interface motion cannot be well approximated by a 1-D Riemann approximation. The reason for this is that near the interface the components of the solution normal to the interface interact strongly with the interface through the interface boundary conditions, whereas the tangential components have a weak, if any, interaction with the interface. Therefore, away from interactions, applying the 1-D jump conditions normal to the interface is a valid approximation. Near an interaction, however, the solution cannot be split into components that are normal and tangential to both interfaces simultaneously and the splitting algorithm is no longer valid.

Simple material interfaces which move with the underlying fluid velocity are not a problem, but for more complicated interactions, such as an oblique shock reflection, then a local grid refinement may be necessary to reduce the splitting errors.

These hybrid methods, combining and interface tracking algorithm with local grid refinement, are also excellent when there is a boundary layer in the solution adjacent to the interface. This might be caused by the energy generated in a flame front or governed by an internal mixing length. If each region is being separately solved on an interface-fitted curvilinear coordinate system, then standard adaptive methods can be used to resolve the boundary layer [2, 24].

Because of the complexity of handling interactions and adaptively refining the interface in 3-D, to my knowledge, the only surface tracking codes that resolve multiple interactions are in 1- and 2-D. In most 2-D codes with interface interactions, and in all 3-D codes I know of, the interfaces are tracked by a volume tracking method.

3. Volume tracking methods

The volume tracking methods are less capable than the surface tracking methods in providing subgrid-scale resolution, but they can simply and accurately account for the interactions of many different smoothly varying interfaces. Here, the interface tracking equations have the same dimension as the underlying PDEs and, therefore, are potentially more expensive than the surface tracking methods. In practice, however, the interface data need only be stored and the equations solved in the cells along or near the interface. This tactic increases the computational complexity but improves the efficiency when the interfaces are well separated.

3.1. *Marker and cell*

One of the earliest volume tracking methods for material interfaces is the marker and cell (MAC) method [29]. Marker particles are scattered initially to identify each material region in the calculation. These particles are transported in a Lagrangian manner along with the materials. Their presence in a computational cell indicates the presence of the marked material. The material boundary is reconstructed using the marker particle densities in the mixed cells with marker particles of two or more materials. The interface reconstruction scheme may also use the density of particles in the surrounding cells to reconstruct a more accurate interface location. A moderate number of particles must be in the mixed cells to reconstruct an accurate interface. If there are only a few marker particles, the interface location will be poorly defined and sensitive to small errors. This often happens in expansion regions unless new particles are continually added and deleted during the calculation.

To improve the efficiency of the MAC method, we can scatter initial marker particles more densely (or only) near the interfaces or, if there are only two materials, only use marker particles to identify one of the materials.

The MAC methods do not require special logic for colliding surfaces but need many marker particles per computational cell to get a well-defined interface. Also, numerical errors in transporting the marker particles can cause an artificial numerical diffusive mixing near the interface resulting in a fuzzy interface. The fuzzy interface makes it harder to generalize the MAC methods for complicated interfaces such as detonations, flame fronts or multiphase flow problems. To reduce the fuzziness far more marker particles than computational cells are needed, increasing the computational cost. The fractional marker volume methods were developed to overcome these drawbacks.

3.2. *Fractional marker volumes*

The fractional marker volume methods (sometimes called the volume of fluid, VOF [14, 15, 21, 22], or simple line interface calculation, SLIC [23], methods) define the surface by calculating the fractional volume of each material occupied in each computational cell. These numbers range from zero (no material) to one (completely filled with material). The interfaces occur in the cells with fractional volumes (fig. 2).

The volume fractions are updated during the calculation according to the appropriate advection equations. On each time step, the interface position is reconstructed cellwise using the fractional volume of a cell and its nearest neighbors. This localness is especially good for long thin interfaces or fingers. When interfaces collide, the fraction volumes are added, and the interface intersections are simply accounted for in the reconstruction step.

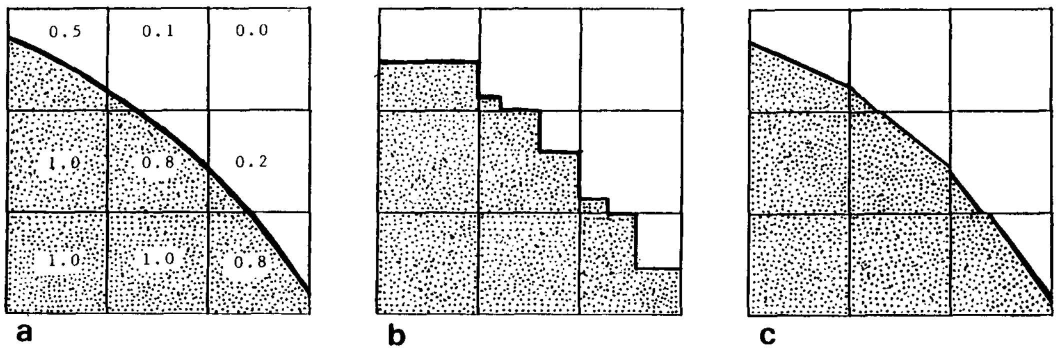

Fig. 2. The original interface separating two regions and the associated volume fractions in each computational cell are shown (2a). In figs. 2b and 2c are two possible reconstructed interfaces using the rectangular and piecewise-linear fractional volume methods, respectively.

3.3. *Interface reconstruction*

Even though the volume fraction in the interface cells are between zero and one, the reconstructed discontinuity is sharp. Within each cell, the volume regions can be represented by unions of rectangles, triangles, or regions bounded by piecewise-polynomial surfaces. Thin rectangles can be used to accommodate fingers. The curvature of the interface can be estimated using finite-difference approximates based on the neighboring fractional volumes. Although the more complicated methods yield a better approximation to the position of the interface, they are more prone to numerical diffusion and nonphysical mixing caused by small pieces shedding off the corners of the reconstructed interface. Imposing monotonicity and convexity constraints [16, 28] on the reconstructed interface greatly reduces the shedding problems.

There are as many fractional volume reconstruction schemes as there are practitioners [1, 3, 4, 14, 15, 20, 21, 23, 26]. The best reconstruction algorithm depends upon the application and the importance of subgrid scale structure. For example, Chorin [3] modified the original SLIC algorithm by allowing for multiple rectangles in a cell and developed a SLICer one for flame propagation. Barr and Ashurst [1] then combined the ideas of slope determination in the VOF method with Chorin's modifications and have one of the SLICest methods in use for turbulent flame propagation.

Another variation of the fractional volume method is to maintain and evolve a point on the interface in each fractional volume cell to aid the reconstruction algorithm. These marker points are moved with the interface during the evolution step. After each reconstruction, the multiple marker points within a single cell are combined to form a single new point on the reconstructed interface, and new marker points are added along the interface in cells with no marker particles. This approach is similar to the reconnecting dual mesh method discussed in the next section.

If the underlying mesh is not a tensor product mesh, then the mapping method can be used to reconstruct the interface using fractional volumes in the uniform logical reference mesh. The curvilinear mesh in the original domain could be chosen adaptively to accurately resolve the solution [2, 18]. The fractional volumes of the reconstructed interface are easily transferred from one mesh to another after a static rezone.

3.4. *Interface evolution*

Once constructed, the interface can be advanced using the same techniques as described for the surface tracking methods. More often, however, they are advanced using a fractional step method. That is, the interface is reconstructed and evolved in each spatial dimension separately. The frac-

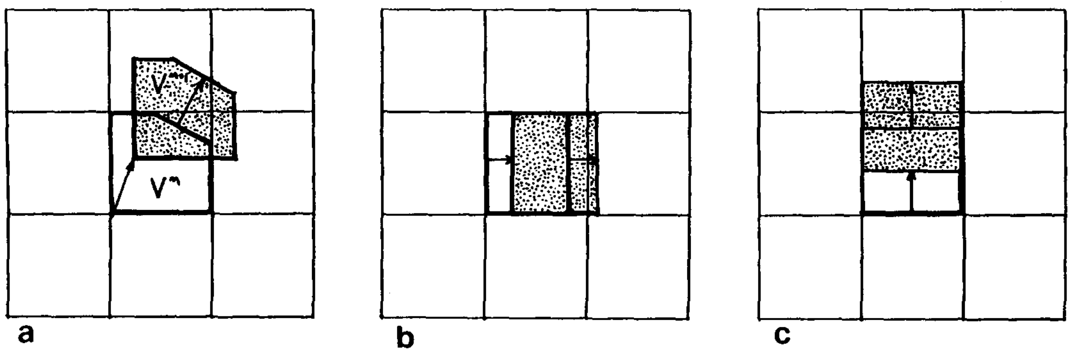

Fig. 3. The reconstructed interfaces for an unsplit algorithm (a), in the x-sweep (b), and in the y-sweep (c).

tional step algorithms are more attractive than the surface tracking methods for almost all interacting surfaces in 2-D and, more so, in 3-D calculations. The unsplit fractional volume method is more difficult to implement because of the odd shaped regions that must be accounted for, such as the one shown in fig. 3a.

In a fractional step method, the interface reconstructed in the x-sweep may be different from what it is in the y-sweep, (figs. 3b,c). By alternating the sweep direction, averaging the results, or using conservative limiters that preserve the symmetry, much of the sweep dependence can be reduced.

4. Moving mesh methods

4.1. *Local adjustment methods*

When using either a surface or volume tracking method, if the reconstructed interface is continuous, then on each time step a local adjustment in the underlying mesh location can be used to approximate the interface. Each mesh point within one half mesh spacing of the interface is moved to the nearest location horizontally or vertically where the interface intersects a mesh line. The interface is now well approximated by the cell edges and diagonals on the new mildly distorted mesh, as seen in the example in fig. 4.

The PDEs are then solved on the new grid treating the mesh points on the interface as a moving boundary; the boundary conditions are determined by the appropriate jump conditions. The correct solution values for the possibly multivalued solution along the interface are easily selected. Also, the discrete approximations to the equations at the few irregular mesh points near the interface are easily derived on each time step [19]. If the finite volume method is being used, the solution jumps across the interface are taken into account in the surface integrals. In many PDEs, accurate and efficient global solution algorithms are available if the discontinuities occur only along mesh boundaries and diagonals [5].

4.2. *Lagrangian methods*

The next logical step is not to have a separate algorithm to track the interface, but to subdivide the initial regions into discrete elements with the cell boundaries aligned with the interfaces. The PDEs are then solved cell by cell, and the cell edges are all treated like interfaces and, therefore, will continue to track the interface. As new interfaces are created, new cells are added (or existing ones readjusted); as the volume of existing cells goes to zero, they are combined or deleted. In this manner, numerical mixing between the different regions is avoided, and varying size structures that move with the solution are easily tracked.

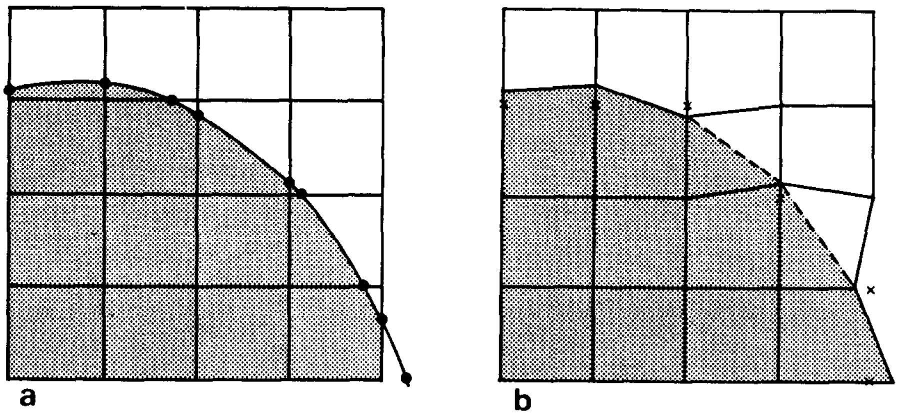

Fig. 4. A simple local adjustment of the mesh aligns it with the front, so the interface is located only along cell edges or diagonals. a) The original underlying mesh and interface; b) the locally adjusted mesh is aligned with the interface.

Depending upon the goal of the calculation, any one of several moving-mesh methods can be used away from the interface [6–10, 13, 17]. Unfortunately, unless the moving cells are allowed to change their nearest neighbors in a calculation, then continual rezoning is necessary to prevent the mesh distortion, in even the simplest flows, from destroying the accuracy of the calculation. This is shown for the very early stages of a Lagrangian Rayleigh–Taylor calculation on a logically rectangular grid in fig. 5. Soon after the time of this plot, numerical errors caused the grid lines to cross and the calculation to fail.

The grid distortion can be somewhat alleviated if only the mesh points along specified interfaces are required to move with the interface. The grid points away from the interface are distributed to prevent mesh tangling or to better resolve the structure of the smooth solution. In addition, the mesh points along the interface can be redistributed along the interface using the surface tracking rezone methods described earlier.

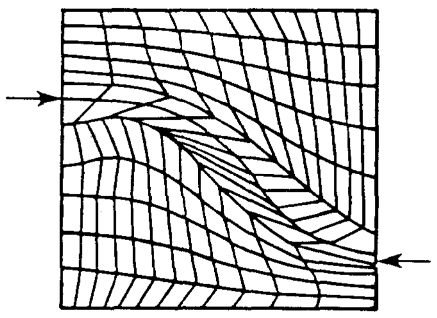

Fig. 5. Distorted Lagrangian mesh in the early stages of a Rayleigh–Taylor instability calculation.

The mesh tangling is not caused by the Lagrangian equations but is an artifact of the mesh data structure. A more flexible data structure that prevents mesh tangling is the neighborhood grid, where pointers are kept to all the nearest neighbors of each mesh point. These are used to approximate the equations locally using a finite element or finite volume discrete approximation. As the mesh moves, the nearest neighbor pointers are continually updated. Usually these reconnections alone will not completely solve the problem of grid resolution and regularity, and some form of rezoning is till necessary. But here, adding and deleting points is much easier than on a logically rectangular or cuboid mesh.

There has been considerable success in using the reconnecting moving neighborhood mesh in fluid flows with material interfaces [7–9, 27]. These free Lagrangian codes, as they are called, could easily be adapted for more complicated flows where the interface moves with the shock or flame velocity rather than with just the underlying fluid velocity.

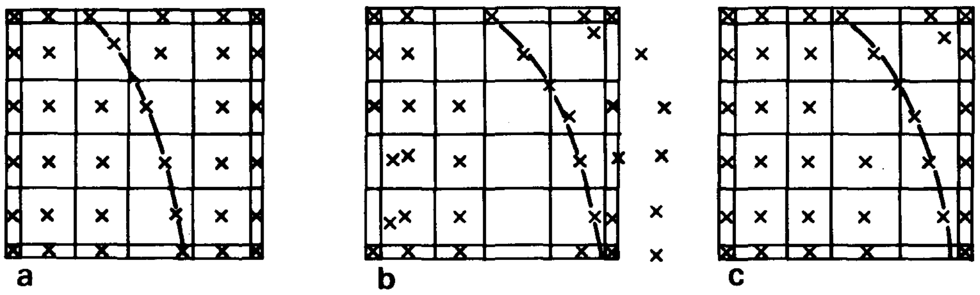

Fig. 6. The dual computing mesh is advanced according to the appropriate moving mesh equations. At the end of each time step it is regularized as in 6c. a) The reference mesh and the dual computing mesh; b) the new dual mesh after one time step; c) the regularized dual mesh at the new time.

The major disadvantage of neighborhood meshes is the data structure. Most of the fast and accurate numerical methods (and multidimensional plotting packages) are for logically rectangular and cuboid grids. The accuracy of local discrete approximations of second derivative operators is particularly poor on neighborhood grids.

One of the more promising reconnecting mesh algorithms retains a logically rectangular or cuboid data structure by approximating the solution on the dual mesh of a logically rectangular or cuboid reference mesh [13, 17]. The dual mesh consists of one mesh point within each cell of the reference mesh (fig. 6a).

At the beginning of every time step, the mesh is regular (one grid point per reference cell). The dual mesh is advanced (fig. 6b) according to the appropriate moving-mesh equations for the interface depicted by a solid line in fig. 6. If desired, an underlying reference mesh can now be chosen to resolve the solution. The mesh is then regularized (fig. 6c) by adding new dual mesh points to empty cells or combining them in reference cells with more than one dual mesh point. In this way, the mesh points where the solution is computed can continue changing their nearest neighbors while maintaining a logically rectangular data structure.

In addition, the dual mesh computing points can be tagged as special interface points (fig. 7a). In fig. 7b the dual mesh points, separating materials □, × and ○, have been advanced forming some new mixed cells in the center. In the regularization stage (fig. 7c) these are combined conservatively to form two new triple points.

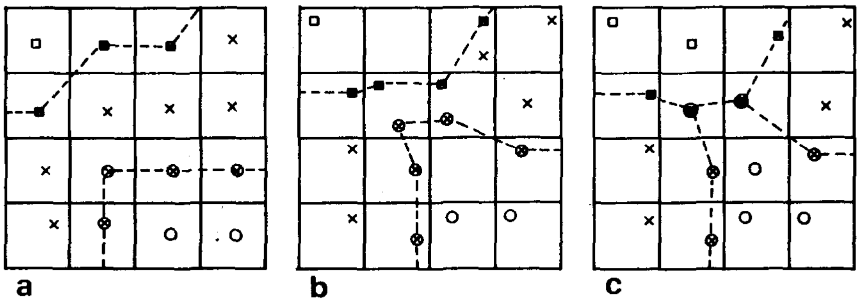

Fig. 7. Interface tracking on the dual mesh. a) Original dual mesh marker points and interface; b) predicted new dual mesh points; c) regularized dual mesh and interface.

5. Summary and conclusions

Most numerical methods for tracking interfaces are based on either following the surface of the interface using marker points, calculating the fractional volume of each region separated by the interfaces as they pass over a reference grid, or moving control volumes aligned with the interface.

The interface is advanced by solving a compatibility equation, usually derived from an integral formulation of the weak form of the PDEs. The smooth solution away from the interface is treated with standard finite-difference or finite-element methods. The interaction of the smooth solution with the interface is accounted for by treating the interfaces as a moving boundary with boundary conditions determined by the compatibility equations.

Reliable methods are available that approximate multiple interactions in 1-D, simple interactions in 1- and 2-D, and isolated interfaces in 1-, 2-, and 3-D. Better methods are still needed for interacting interfaces in 2- and 3-D. It is not yet clear which of the existing methods will prove to be the most effective in these complicated situations.

Acknowledgements

I thank Andy White for pointing out the distinction between a numerical method and a numerical scheme.

References

[1] P.K. Barr and Wm.T. Ashurst, An Interface Scheme for Turbulent Flame Propagation, Sandia National Laboratories report SAND82-8773 (1982).
[2] J.U. Brackbill and J.S. Saltzman, Adaptive Zoning for Singular Problems in Two Dimensions, J. Comp. Phys. 46 (1982) 342–368.
[3] A.J. Chorin, Flame Advection and Propagation Algorithms, J. Comp. Phys. 35 (1980) 1–11.
[4] P. Colella, P. Concus and J.A. Sethian, Some Numerical Methods for Discontinuous Flows in Porous Media, Lawrence Berkeley Laboratory report LBL-15932, April 1983.
[5] J.E. Dendy and J.M. Hyman, Multigrid and ICCG for Problems with Interfaces, Elliptic Problem Solvers, Martin Schultz, ed. (Academic Press, New York, 1981), pp. 247–254.
[6] J.K. Dukowicz, A Simplified Adaptive Mesh Technique Derived from the Moving Finite Element Method, Los Alamos National Laboratory report LA-UR-82-3664.
[7] J.K. Dukowicz, Lagrangian Fluid Dynamics using the Voronoi–Delaunay Mesh, Los Alamos Scientific Laboratory report LA-UR-81-1421.
[8] M.J. Fritts, Lagrangian Simulations of the Kelvin–Helmholtz Instability, Science Applications Inc. report SAI-76-632-WA, September 1976.
[9] M.J. Fritts and J.P. Boris, The Lagrangian Solution of Transient Problems in Hydrodynamics using a Triangular Mesh, J. Comp. Phys. 31 (1979) 173–215.
[10] R.J. Gelinas and S.K. Doss, The Moving Finite Element Method: Applications to General Partial Differential Equations with Multiple Large Gradients, J. Comp. Phys. 40 (1981).
[11] J. Glimm, E. Isaacson, D. Marchesin and O. McBryan, Front Tracking for Hyperbolic Systems, Adv. Appl. Math. 2 (1981) 91–119.
[12] J. Glimm, D. Marchesin and O. McBryan, A Numerical Method for Two Phase Flow with an Unstable Interface, J. Comp. Phys. 39 (1981) 179–200.
[13] A. Harten and J.M. Hyman, A Self-Adjusting Grid for the Computation of Weak Solutions of Hyperbolic Conservation Laws, J. Comp. Phys. 50 (1983) 235–269.
[14] C.W. Hirt and B.D. Nichols, A Computational Method for Free Surface Hydrodynamics, American Society of Mechanical Engineers 80-C2/PVP-144, April 1980.
[15] C.W. Hirt and B.D. Nichols, Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries, J. Comp. Phys. 39 (1981) 201–225.
[16] J.M. Hyman, Accurate Monotonicity Preserving Cubic Interpolation, Los Alamos Report No. LA-8796-MS, to be published in SIAM J. Sci. and Stat. Comp.
[17] J.M. Hyman, Adaptive Moving Mesh Methods for Partial Differential Equations, Advances in Reactor Computations (American Nucl. Soc., La Grange Park, IL, 1983) pp. 24–43.
[18] J.M. Hyman, Adaptive Static Rezoning Methods, to be published in the proceedings AMS-SIAM Conference on Large Scale computations in Fluid Mechanics (1983).
[19] J.M. Hyman and B. Larrouturou, The Numerical Differentiation of Discrete Functions Using Polynomial Interpolation Methods, Appl. Math. and Comp., vols. 10–11 (1982) pp. 487–506.
[20] P. Lötstedt, A Front Tracking Method Applied to Burgers' Equation and Two-Phase Porous Flow, J. Comp. Phys. 47 (1982) 211–228.
[21] B.D. Nichols, C.W. Hirt and R.S. Hotchkiss, SOLA-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries, Los Alamos Scientific Laboratory Report LA-8355, August 1980.
[22] B.D. Nichols and C.W. Hirt, Methods for Calculating Multi-Dimensional, Transient Free Surface Flows Past Bodies, Los Alamos Scientific Laboratory report LA-UR-

75-1932; Proc. First Int. Conf. on Numerical Ship Hydrodynamics, Gaithersburg, MD, October 1975.

[23] W.F. Noh and P. Woodward, SLIC (Simple Line Interface Calculation), in Lecture Notes in Physics, vol. 59: Proc. Fifth Int. Conf. on Numerical Methods in Fluid Dynamics, A.I. van de Vooren and P.J. Zandbergen, eds. (Springer-Berlin, 1976).

[24] J.F. Thompson, ed., Numerical Grid Generation (North-Holland, Amsterdam, 1982).

[25] E.A. Overman II and N.J. Zabusky, Evolution and Merger of Isolated Vortex Structures, Inst. for Computational Mathematics and Applications Report ICMA-81-31 (1981).

[26] J. Sethian, Turbulent Combustion in Open and Closed Vessels, Lawrence Berkeley National Laboratory report LBL-15744, Feb. 1983.

[27] H.E. Trease, A Two-Dimensional Free Lagrangian Hydrodynamics Model, Ph. D. Thesis, Sept. 1981.

[28] Bram van Leer, Towards the Ultimate Conservative Difference Scheme. V. A Second-Order Sequel to Godunov's Method, J. Comp. Phys. 32 (1979) 101–136.

[29] J.E. Welch, F.W. Harlow, J.P. Shannon and B.J. Daly, The MAC Method: A Computing Technique for Solving Viscous, Incompressible, Transient Fluid Flow Problems Involving Free Surfaces, Los Alamos Scientific Laboratory report LA-3425, 1966.